CHAPTER

3

The UNIX Editors

case ▶ All employees at Dominion Consulting write weekly memos summarizing their activities and accomplishments. They write these memos using a UNIX text editor, which lets you create and modify simple text-based documents. UNIX includes at least two text editors—vi editor and Emacs. As part of your UNIX training, your manager at Dominion Consulting, Rolfe Williams, asks you to use these editors to create two versions of the same memo summarizing basic features of text files.

LESSON A

objectives

After studying this lesson, you should be able to:

- **Describe an ASCII text file**
- **Explain why operating system editors use ASCII files**
- **Create and edit simple documents using the vi editor**

The vi Editor

This chapter introduces two UNIX editors. An **editor** is a program for creating and modifying computer documents such as programs and data files. Files may contain notes, memos, or program source code, for example. A **text editor** is like a simplified word processor; you can use a text editor to create and edit documents, but you cannot format them using boldfaced centered text or other features. All operating systems have a standard editor; many also include alternate editors. In UNIX, the standard editor is the vi editor and the most popular alternate is the Emacs editor.

Understanding UNIX Files

Almost everything you create in UNIX is stored in a file. All information stored in files is in the form of binary digits. A **binary digit**, called **bit** for short, consists of two numbers, 0 and 1. Because the computer consists of electronic circuits that are either in an "on" or "off" state, binary numbers are perfectly suited to report these states. The exclusive use of 0s (which mean "off") and 1s (which mean "on") as a way to communicate with the computer is known as **machine language.** The earliest programmers had to write their programs using machine language, a tedious and time-consuming process.

ASCII Text Files

To make information stored in files accessible, computer designers established a standard method for translating binary numbers into plain English. This standard used a string of eight binary numbers, called a **byte,** which is an acronym for "binary term." A byte can be configured into fixed patterns of bits, and these patterns can be interpreted as an alphabetic character, decimal number, punctuation mark, or a special character, such as &, *, or @. Each byte, or **code**, has been standardized into a set of bit patterns known as ASCII codes. **ASCII** stands for the American Standard Code for Information Interchange. Computer files containing nothing but ASCII characters are called **text files**, and files that contain non-ASCII characters, such as machine instructions, are called **binary files**. Figure 3-1 lists the ASCII characters.

Printing Characters (Punctuation Characters)

Dec	*Octal*	*Hex*	*ASCII*
32	040	20	(Space)
33	041	21	!
34	042	22	“
35	043	23	#
36	044	24	$
37	045	25	%
38	046	26	&
39	047	27	‘
40	050	28	(
41	051	29	)
42	052	2A	*
43	053	2B	+
44	054	2C	,
45	055	2D	-
46	056	2E	.
47	057	2F	/

(Decimal Numbers—Print)

Dec	*Octal*	*Hex*	*ASCII*
48	060	30	0
49	061	31	1
50	062	32	2
51	063	33	3
52	064	34	4
53	065	35	5
54	066	36	6
55	067	37	7
56	070	38	8
57	071	39	9

(Special Characters—Print)

Dec	*Octal*	*Hex*	*ASCII*
58	072	3A	:
59	073	3B	;
60	074	3C	<
61	075	3D	=
62	076	3E	>
63	077	3F	?
64	080	40	@

Printing Characters (Alphabet—Upper Case)

Dec	*Octal*	*Hex*	*ASCII*
65	101	41	A
66	102	42	B
67	103	43	C
68	104	44	D
69	105	45	E
70	106	46	F
71	107	47	G
72	110	48	H
73	111	49	I
74	112	4A	J
75	113	4B	K
76	114	4C	L
77	115	4D	M
78	116	4E	N
79	117	4F	O
80	120	50	P
81	121	51	Q
82	122	52	R
83	123	53	S
84	124	54	T
85	125	55	U
86	126	56	V
87	127	57	W
88	130	58	X
89	131	59	Y
90	132	5A	Z

Printing Characters (Alphabet—Lower Case)

Dec	*Octal*	*Hex*	*ASCII*
97	141	61	a
98	142	62	b
99	143	63	c
100	144	64	d
101	145	65	e
102	146	66	f
103	147	67	g
104	150	68	h
105	151	69	i
106	152	6A	j
107	153	6B	k
108	154	6C	l
109	155	6D	m
110	156	6E	n
111	157	6F	o
112	160	70	p
113	161	71	q
114	162	72	r
115	163	73	s
116	164	74	t
117	165	75	u
118	166	76	v
119	167	77	w
120	170	78	x
121	171	79	y
122	172	7A	z

Non-Printing Characters (Abridged)
Control Characters

Dec	*Octal*	*Hex*	*ASCII*
0	000	00	^@ (Null)
7	007	07	Bell
8	010	08	Backspace
9	011	09	Tab
10	012	0A	LineFeed, Newline
11	013	0B	Vertical Tab
12	014	0C	Formfeed
13	015	0D	Carriage return

Figure 3-1: ASCII characters

GUI Files

Computers are not limited to processing ASCII codes. To work with graphic information, such as icons, illustrations, and other images, binary files can include strings of bits representing white and black dots, where each black dot represents a 1 and each white dot a 0. Graphics files include bit patterns—rows and columns of dots called a **bitmap**—that must be translated by graphics software, commonly called a **GUI** (Graphical User Interface), which transforms a complex array of bits into an infinite variety of images.

Executable Program Files

Many programmers develop source code for their programs by writing text files; then they compile these files to convert them to executable program files. **Executable program files** contain pure binary or machine language that the computer can immediately use or execute.

Using Operating System Editors

Operating system editors let you create and edit simple ASCII files. UNIX includes two editors: vi and Emacs. They are **screen editors**: they display the text you are editing one screen at a time and let you move around the screen to change and add text. You can also use a line editor to edit text files. A **line editor** lets you work with only one line or group of lines at a time. Although line editors do not let you see the context of your editing, they are useful for general tasks such as searching, replacing, and copying blocks of text.

Using the vi Editor

The vi editor is so called because it is visual—it immediately displays on screen the changes you make to text. It is also a **modal editor**; that is, it works in two modes: insert mode and command mode. **Insert mode** lets you enter text; **command mode** lets you enter commands to perform editing tasks, such as moving through the file and deleting text.

Now you're ready to write your first memo using the vi editor. To do so, you complete the following tasks:

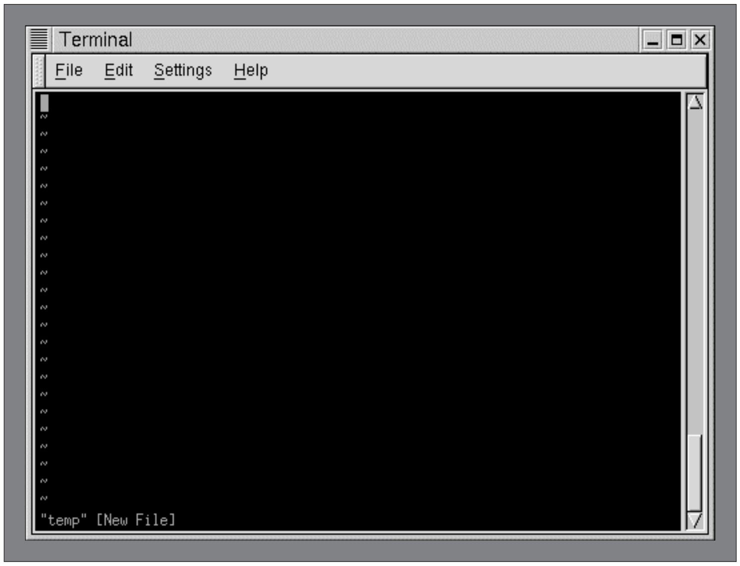
- Create a new file in the vi editor
- Insert, edit, and delete text
- Search and replace text
- Add text from other files
- Copy, cut, and paste text
- Print a file
- Save a file and exit vi

Creating a New File in the vi Editor

Start by creating a file called temp to hold your first memo summarizing the basic features of text files. To start vi and create a file simultaneously, you type vi followed by the new file's name.

To enter vi and create a new file:

- After the $ command prompt, type **vi temp** and press **Enter**. This starts vi and creates a new file called temp. Your screen should look similar to Figure 3-2.



```
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"temp" [New File]
```

Figure 3-2: vi editor's opening screen

In the upper-left corner of your screen, you see the cursor, shown in Figure 3-2 as an underline character. The cursor indicates your current location in the file.

Note: The line containing the cursor is the current line. Lines containing tildes (~) are not part of the file: they indicate lines on the screen only, not lines of text in the file.

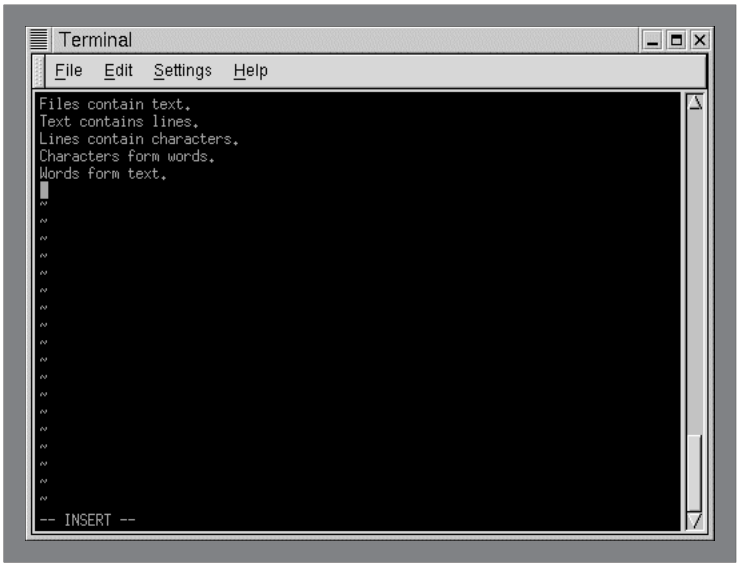
Inserting Text

When you start the vi editor, you're in command mode. This means that the editor interprets anything you type on the keyboard as a command. Before you can insert text in your new file, you must use the i (insert) command.

To insert text in the Temp file:

1. Type **i**.

   Like most vi commands, the i command does not appear (or echo) on your screen. The command switches you from command mode to insert mode; you don't need to press Enter to signal the command's completion.
2. Type the text shown in Figure 3-3.



```
Files contain text.
Text contains lines.
Lines contain characters.
Characters form words.
Words form text.

~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --
```

Figure 3-3: Inserting text with the vi editor

If you need to delete characters, press Backspace. Press Enter at the end of each line to move to the next line.

Note: In insert mode, every character you type appears on the screen. To switch to command mode, press Esc until "INSERT" is no longer displayed.

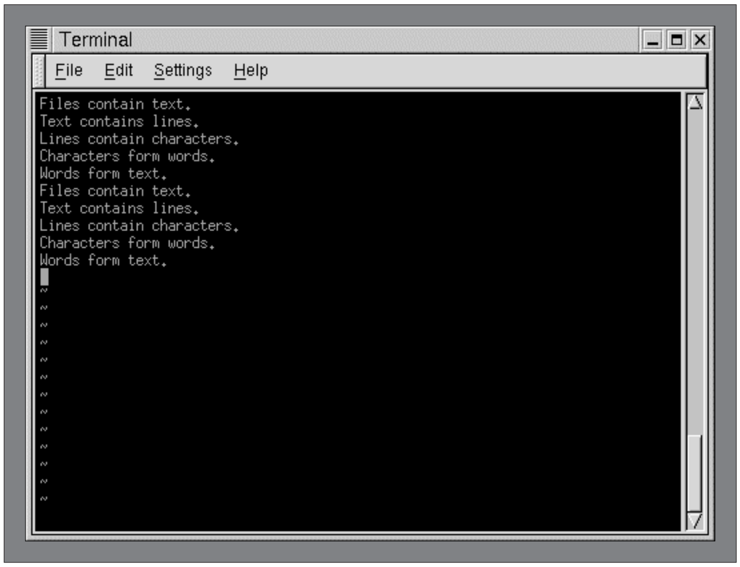
Repeating a Change

Use the repeat (**.**) command to repeat the most recent change you made. Because you just inserted text, the repeat command repeats the insertion, duplicating the inserted text. You can add the next item in your memo by repeating the previous line and then editing it.

To repeat your last command:

1. Press **ESC** to switch to command mode.
2. Type **.** (period).

   The vi editor inserts the last text you typed below the current line. Your screen should look similar to the one in Figure 3-4.



```
Files contain text.
Text contains lines.
Lines contain characters.
Characters form words.
Words form text.
Files contain text.
Text contains lines.
Lines contain characters.
Characters form words.
Words form text.

~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

Figure 3-4: Repeating a command in the vi editor

Now you're ready to edit the text you just inserted. Start by moving the cursor around the screen and then deleting unnecessary text.

Moving the Cursor

You can move the cursor when you are in command mode. Start by moving around the screen to get a feel for the commands, and then move to the particular line you want to edit.

To move the cursor around the screen:

1. Press **ESC** to make sure you are in enter command mode.
2. Press the arrow keys to move up, down, left, and right one character at a time.
3. Type **H** to move the cursor to the upper-left corner of the screen.

   Note: Make sure you type capital letters as indicated in these steps.
4. Type **L** to move the cursor to the last line on the screen.
5. Type **G** to go to the beginning of the last line. This is the go to command. You can include a number before the G to indicate which line you want to move to.
6. Type **2G** to move to the beginning of the second line.

In addition to these commands, you can use other commands to move the cursor. Table 3-1 summarizes the vi editor's cursor movement keys.

Key	Movement
h or left arrow	Left one character position
l or right arrow	Right one character position
k or up arrow	Up one line
j or down arrow	Down one line
H	Upper-left corner of the screen
L	Last line on the screen
*n*G	Go to the line specified by a number, *n*
w	Forward one word
b	Back one word
0 (zero)	To the beginning of the current line
$	To the end of the current line
Ctrl+U	Up one-half screen
Ctrl+D	Down one-half screen
Ctrl+F or Pg down	Forward one screen
Ctrl+B or Pg up	Back one screen

Table 3-1: vi editor's cursor movement keys

Remember that cursor movement keys only work in command mode.

Note: Using the letter keys to move the cursor dates back to the days when UNIX used teletype terminals that had no arrow keys. Designers of vi chose the letter keys because of their relative position on the keyboard.

Deleting Text

Now that you know how to insert text and move around a file, you are ready to delete text. To do so, move to a character and then type x to delete that character. You can also combine many delete commands with cursor movement commands to delete more than one character. Table 3-2 summarizes the most common delete commands.

Command	Purpose
x	Delete the character above the cursor
dd	Delete the current line
dw	Delete the word above the cursor. If the cursor is in the middle of the word, delete from the cursor to the end of the line.
d$ or D	Delete from the cursor to the end of the line
d0	Delete from the cursor to the start of the line

Table 3-2: vi editor's delete commands

Now you can use the delete commands and the cursor movement keys to edit text you inserted in your memo.

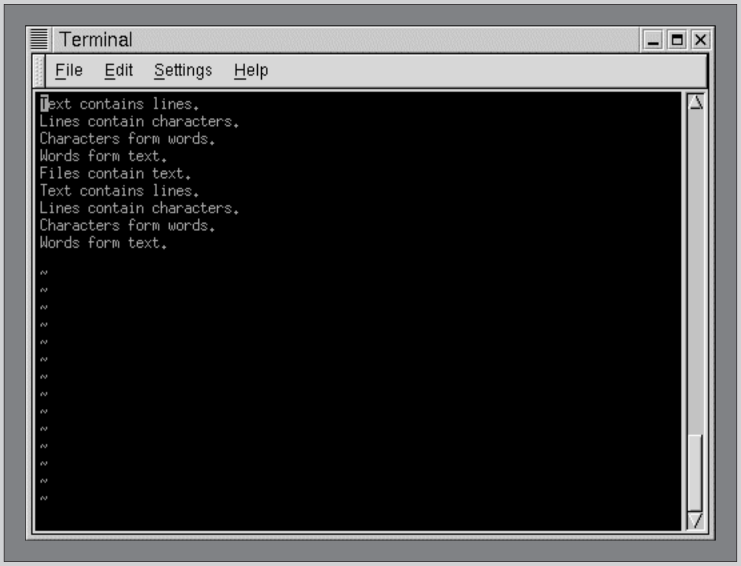
To edit the temp file by deleting text:

1 Press **Esc** to make sure you are in command mode.

2 Type **1G** to move to the first line of the file. You want to delete this line.

3 To delete the first line, type **dd**.

Your file should now look like Figure 3-5.



```
Text contains lines.
Lines contain characters.
Characters form words.
Words form text.
Files contain text.
Text contains lines.
Lines contain characters.
Characters form words.
Words form text.

~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

Figure 3-5: Memo after deleting first line

4 Press **w** to go to the next word, "contains."

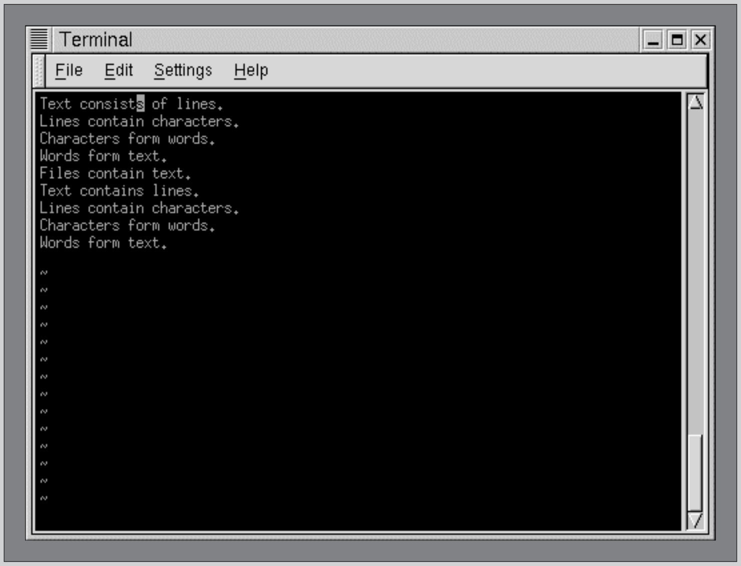
5 Type **dw** to delete the current word (so the line now reads "Text lines"), and then type **i** to enter insert mode.

6 Type **consistss of** between "Text" and "lines." Be sure to include the extra "s."

7 Press the arrow keys to move the cursor to the extra "s" in "consistss," and then press **ESC** to switch to command mode.

8 To delete the current character (the extra "s"), type **x**.

Your memo should now look like the one in Figure 3-6.



```
Text consists of lines.
Lines contain characters.
Characters form words.
Words form text.
Files contain text.
Text contains lines.
Lines contain characters.
Characters form words.
Words form text.
~
~
~
~
~
~
~
~
~
~
~
~
~
~
```

Figure 3-6: Memo after editing

Now you want to edit the sentence, "Files contain text" by deleting the last word.

9 Press the arrow keys to move to the sentence, "Files contain text," and then move to the first "c" in "contain."

10 Type **d$** to delete the text from the cursor to the end of the line, and then type **i** to switch to insert mode.

11 Type **consist of words.** to complete the sentence.

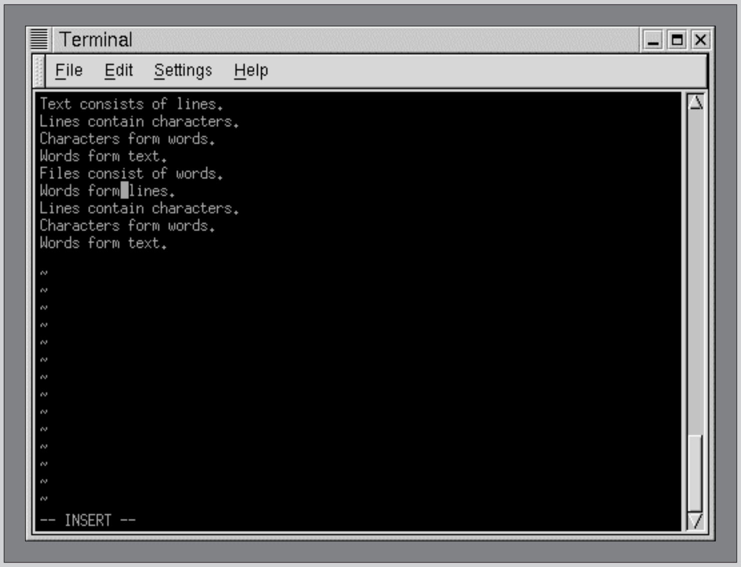
Now you can edit the next sentence by replacing the first word.

12 Press the arrow keys to move to the next line in the file, move to the space before the word "lines," and then press **ESC** to switch back to command mode.

13 Type **d0** to delete the text from the cursor to the beginning of the line, and then type **i** to enter insert mode.

14 Type **Words form** to insert that text at the beginning of the sentence.

15 Your completed memo should look like the one in Figure 3-7.



```
Text consists of lines.
Lines contain characters.
Characters form words.
Words form text.
Files consist of words.
Words form lines.
Lines contain characters.
Characters form words.
Words form text.

~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --
```

Figure 3-7: Completed memo after editing

The vi editor offers you alternatives for copying, cutting, and pasting text.

Note: The delete line command, dd, actually places deleted lines in a buffer. Then you can use the paste command, p, to paste deleted (cut) lines elsewhere in the text. (Position the cursor where you want to paste them.) To copy and paste text, use the yank command, yy, to copy the lines. After yanking the lines you want to paste elsewhere, move the cursor and type p to paste the text in the current location.

Undoing a Command

If you complete a command and then realize you want to reverse its effects, you can use the undo command. For example, if you delete a few lines from a file by mistake, type u to restore the text. The undo command reverses only your last command.

Searching for a Pattern

You can search forward for a pattern of characters by typing a forward slash (/), typing the pattern you are seeking, and then pressing Enter. For example, suppose you want to know how many times you used the word “consist” or “consists” in your memo.

To search for a pattern of text to find either word:

1 Press **Esc** to make sure you are in command mode.

2 Type **H** to move the cursor to the top of the screen.

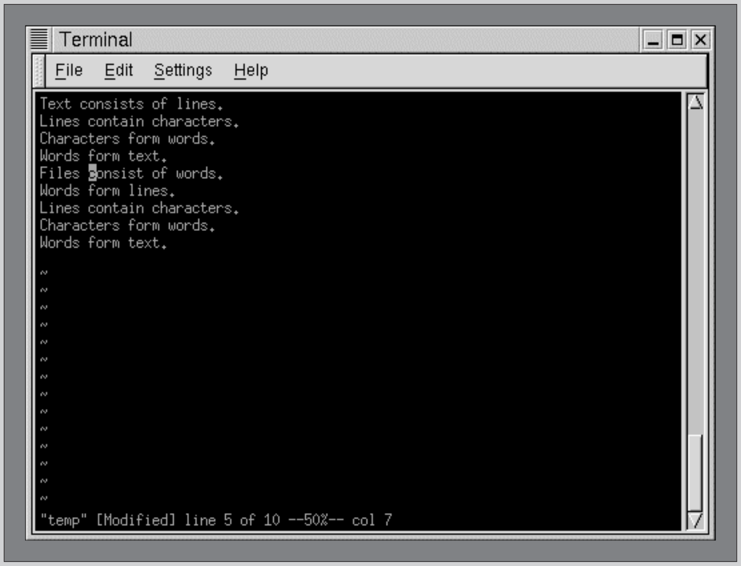
3 Type **/cons** to search for words that start with "cons."

4 Press **Enter**. The cursor moves to the beginning of the word "consists" on line 1.

5 To search for the next occurrence of "cons," press **n** (for next). The cursor moves to the beginning of the word "consist" on the fifth line.

If you had searched for "/con," you would have first found "consist" on line 1 and then "contain" on line 2.

6 To see file status information, press **Ctrl+K**. Your screen should look like the one in Figure 3-8.



```
Text consists of lines.
Lines contain characters.
Characters form words.
Words form text.
Files consist of words.
Words form lines.
Lines contain characters.
Characters form words.
Words form text.
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"temp" [Modified] line 5 of 10 --50%-- col 7
```

Figure 3-8: vi status line

Note: The status line at the bottom of the screen displays information, including line-oriented commands (explained later in this lesson) and error messages.

Searching and Replacing

Let's say you want to change all occurrences of "text" in your memo to "documents." Instead of searching for "text" and then deleting and inserting "documents," you can search and replace with one command. The commands you learned so far are screen-oriented. Commands that can perform more than one action (searching and replacing) are **line-oriented commands**.

Note: Screen-oriented commands execute at the location of the cursor. You do not need to tell the computer where to perform the operation: it takes place relative to the cursor. Line-oriented commands, on the other hand, require you to specify an exact location (an address) for the operation. Screen-oriented commands are easy to type, and their changes appear on the screen. Typing line-oriented commands is more complicated, but they can execute independently of the cursor and in more than one place in a file.

A colon (:) precedes all line-oriented commands. It acts as a prompt on the status line. Enter line-oriented commands on the status line, and press Enter when you complete the command.

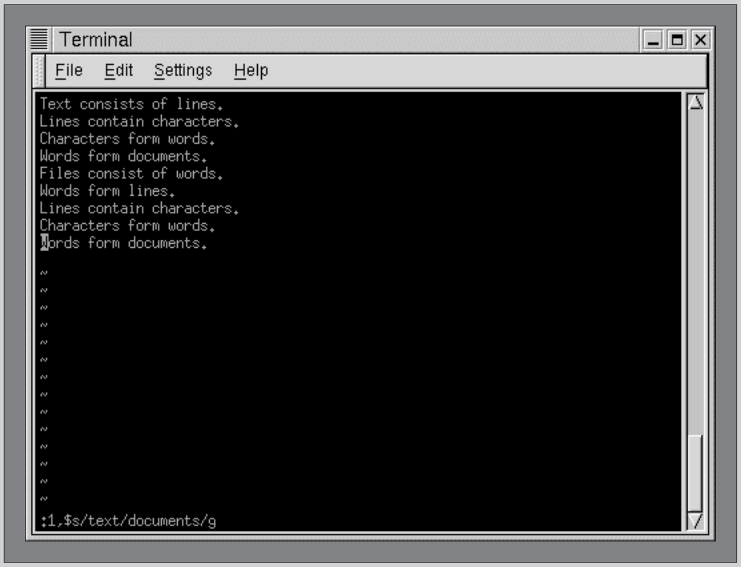
Note: In this chapter, all instructions for line-oriented commands include the colon as part of the command.

To search for a pattern of characters:

1. Press **Esc** to make sure you are in command mode.
2. Type **/form** and press **Enter**. This command instructs vi to search for the first occurrence of the word "form" and moves the cursor under that word.
3. Type **n** to repeat the search. vi locates the next occurrence of the word "form."

To search for "text" and replace it with "documents":

1. Press **Esc** to make sure you are in command mode.
2. Type **:1,$s/text/documents/g**. This command means "From the first line (1) to the end of the file ($), search for 'text' and replace it with 'documents' (s/text/documents/) everywhere it occurs on each line (g)."
3. Press **Enter**. Your screen should look like the one illustrated in Figure 3-9.



```
Text consists of lines.
Lines contain characters.
Characters form words.
Words form documents.
Files consist of words.
Words form lines.
Lines contain characters.
Characters form words.
Words form documents.
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:1,$s/text/documents/g
```

Figure 3-9: Memo after searching and replacing

Note that the word "Text" in line 1 remains unchanged because it is capitalized. By default, case matters in searches.

Saving a File and Exiting vi

As you edit a file, saving your changes is a good idea. You should always save the file before you exit vi; otherwise, you lose your changes.

To save the temp file:

- Type **:x** and then press **Enter** to save your changes, exit the vi editor, and return to the UNIX shell.

  The status line provides information about your file, including its name and the number of lines and characters it contains.

Note: While in command mode, you can also use :wq (write and quit) or ZZ to exit the editor after you save the file on disk. To save a file and continue working with the vi editor, type :w.

Adding Text from Another File

Sometimes the text you want to include in one file is already part of another file. For example, let's say you want a separate copy of the text in the file temp so you can use it to practice editing. Start by creating a new file called practice, and then add text from the temp file by using the line-oriented r (read) command.

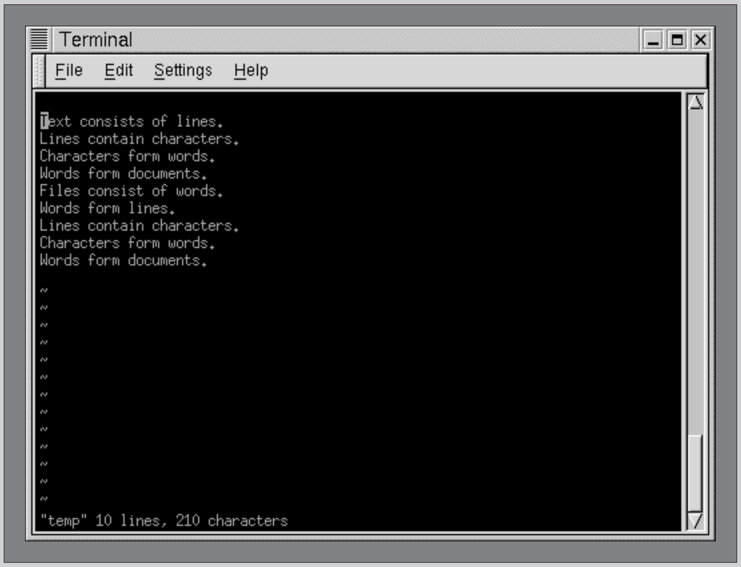
To create a new file and add text from another file:

1 Type **vi practice** and press **Enter** to create a new file.

2 Press **Esc** to make sure you are in command mode.

3 Type **:r temp**.

Your file should look like the one illustrated in Figure 3-10.



```

Text consists of lines.
Lines contain characters.
Characters form words.
Words form documents.
Files consist of words.
Words form lines.
Lines contain characters.
Characters form words.
Words form documents.

~
~
~
~
~
~
~
~
~
~
~
~
~
"temp" 10 lines, 210 characters
```

Figure 3-10: Adding text from another file

The r command copied the text from temp and put it in the current file, practice. Notice the blank line at the top of the file.

4 Move the cursor to the blank line, and type **dd** to delete it.

Leaving vi Temporarily

If you want to execute other UNIX commands while you work with vi, you can leave vi temporarily. For example, let's say you are working on your memo for Dominion and want to check the current date quickly.

To leave vi temporarily to find the current date and time:

1 Type **:!date** and press **Enter**.

You see today's date and instructions for returning to command mode.

2 Press **Enter** to return to command mode.

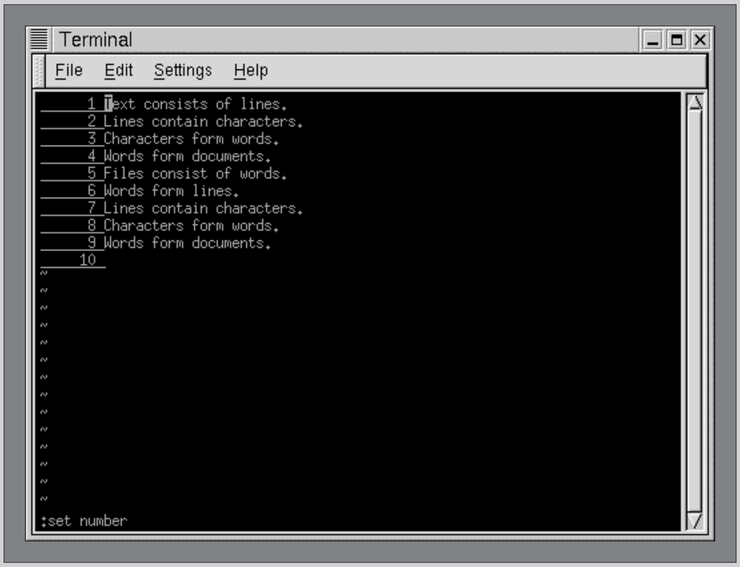
Changing Your Display While Editing

Besides using the vi editing commands, you can also set options in vi to control editing parameters such as line number display and whether case matters in searches. Turn on line numbering when you want to work with a range of lines, for example, when you're deleting or cutting and pasting a block of text. Then you can refer to the line numbers to specify the text. You decide to delete the last three lines from your memo. Turn on line numbering first, and then use a delete command.

To use automatic line numbering:

1 Type **:set number** and press **Enter**.

Your redrawn screen shows line numbers to the left of the text. Your screen should look like the one in Figure 3-11.



```
      1 Text consists of lines.
      2 Lines contain characters.
      3 Characters form words.
      4 Words form documents.
      5 Files consist of words.
      6 Words form lines.
      7 Lines contain characters.
      8 Characters form words.
      9 Words form documents.
     10
~
~
~
~
~
~
~
~
~
~
~
~
~
~
:set number
```

Figure 3-11: Changing your display

Line numbers are for reference only. They are not part of the file. Now you can use these reference numbers to delete the last three lines in the file.

2 Type **:7,9d** and press **Enter**.
You deleted the last three lines of the file.

Copying or Cutting and Pasting

You can use the yy command in vi to copy a specified number of lines from a file and place them on the clipboard. To delete the lines from the file and store them on the clipboard, use the dd command. Now you want to reorganize your document so that the first three lines are at the end of the file. You can use cut and paste commands to make this change.

To cut and paste text:

1 Type **H** to move the cursor to the beginning of line 1.
2 Type **3dd** to cut the first 3 lines from the document and store them on the clipboard.
3 Type **G** to move the cursor to the end of the file.
4 Type **p** to paste the three lines at the end of the file.

Printing vi files

You can use the lpr (line print) shell command to print a file. Type !lpr and then type the name of the file you want to print.

To print a file:

1 Press **Esc** to return to command mode.
2 Type **!lpr practice** and press **Enter**. This prints the file practice in the current directory on the default printer.

You can also specify which printer you want to use with the –Pprinter option. For example, you may use two printers—lp1 and lp2. To print the practice file on lp2, type !lpr –Plp2 practice and press Enter.

Canceling an Editing Session

If necessary, you can cancel an editing session, that is, you can undo all the changes you made. If you are working with a new file, vi deletes the file. If you are modifying an existing file, vi restores its original condition. You decide you don't need your practice file, so you can cancel your editing session.

To cancel your editing session:

- Type **:q!** and press Enter.

  This cancels all changes you made, and because practice is a new file, vi deletes it.

SUMMARY

- Bytes are "computer characters" referred to as "codes." These codes have been standardized and are known as ASCII codes. ASCII stands for the American Standard Code for Information Interchange. Computer files that contain only ASCII characters (bytes) are called text files.
- The vi editor remains the choice of most UNIX users. All operating systems editors process text files. Text files are also called "flat files" or "ASCII files."
- The vi editor is a modal editor, because it works in two modes: insert mode and command mode. Insert mode lets you enter text, and command mode lets you navigate the file and modify the text.
- In the vi editor's insert mode, characters you type are inserted in the file. They are not interpreted as vi commands. To exit insert mode and re-enter command mode, press Esc.
- With vi, you initially edit a copy of the file placed in the computer's memory. You do not alter the file itself until you save it on disk.

COMMAND SUMMARY

Chapter 3, Lesson A commands

Command	Purpose
vi Commands	
. (repeat)	Repeat your most recent change
/	Search forward for a pattern of characters
:!	Leave vi temporarily
:q	Cancel an editing session
:r	Read text from one file and add it to another
:set	Turns on certain options, such as line numbering

Chapter 3, Lesson A commands (continued)	
:w	Save a file and continue working
:wq	Write changes to disk and exit vi
:x	Save changes and exit vi
i	Switch to insert mode
P	Paste text from the clipboard
u	Undo your most recent change
vi	Start the vi editor
yy	Copy (yank) text to the clipboard
ZZ	In command mode, save changes and exit vi
UNIX Commands	
lpr	Print a file

Note: Table 3-1 lists the vi editor's cursor movement keys. Table 3-2 lists the vi editor's delete commands.

REVIEW QUESTIONS

1. The vi editor is called a modal editor because it ____________.
  a. is portable
  b. operates in two modes
  c. makes entering and changing text easy
  d. operates in a single mode

2. Which key do you press to switch from insert mode to command mode?
  a. Esc
  b. :
  c. Ctrl+C
  d. Ins

3. While in command mode, you can exit vi by ____________.
  a. typing :quit!
  b. typing :wq!
  c. typing :x
  d. all of the above

4. Why is the vi editor called a "visual editor?"
  a. Because it works with a line editor
  b. Because it is line-oriented
  c. Because it uses a graphical interface
  d. Because text changes appear immediately on the screen

5. In the vi editor, the colon (:) serves as the ________________.
a. status line command prompt
b. exit command
c. substitution command
d. search command

6. In the vi editor's command mode, what single character do you type to start searching for a text pattern?
a. ?
b. s
c. /
d. :

7. Operating system editors work with ________________ files.
a. executable program
b. flat
c. graphical
d. system

8. Which of these vi commands deletes one line from a file?
a. d$
b. d0
c. dd
d. x

9. Which of these vi commands is a cursor movement command?
a. G
b. l
c. h
d. all of the above

10. Which of these vi commands adds text from another file?
a. r
b. read
c. :r
d. :w

11. Which of these commands displays line numbers in a vi file?
a. set no
b. :set
c. :number
d. :set number

12. Which vi command causes you to leave vi temporarily?
a. :t
b. :wq
c. :!
d. :u

13. Which vi command repeats your most recent change?
a. .
b. :r
c. :!
d. :u

14. Which vi command copies text to the clipboard?
a. :y
b. C
c. :c
d. Y

15. Which vi command pastes text from the clipboard?
a. :p
b. P
c. :!
d. :u

16. Which vi command undoes your most recent changes?
a. :wq
b. u
c. :o
d. Z

17. What does the H command perform?
a. causes the cursor to move to the beginning of line 1
b. causes the cursor to move to the beginning of the current line
c. holds the contents of the file in memory for future editing
d. none of the above

18. What does the command 7dd perform?
a. deletes the seventh word on the current line
b. deletes seven lines
c. deletes the third line from the current line
d. none of the above

19. What does the command G perform?
a. grabs text and places it in the clipboard
b. moves the cursor to the first occurrence of the letter G
c. moves the cursor to the end of the file
d. moves the cursor to the end of the current line

EXERCISES

1. Using the vi editor:
   a. Create a document with 10 lines.
   b. Enter the word "these" on every line.
   c. Save the file.
   d. Re-open the document and change "these" to "those" throughout the file.
2. Using the vi editor:
   a. Create a document called first.file, and enter a few lines of text in it. Save it.
   b. Create a second document called second.file, and enter a few lines of text in it. Save it.
   c. Create a third document called third.file, by adding the text from the first two files.
   d. Save the third file and exit from the editor.
   e. Type **vi third.file** to be sure third.file contains the text from both files.
3. Delete all text from third.file, and then restore it.

DISCOVERY EXERCISES

1. Create a file with 10 lines of text. Remove the first four lines, and add them to the end of the file.
2. On a line in the middle of the file, insert your name.
3. Move the cursor to the beginning of the file.
4. Enter the command to search for your first name and replace it with "George." (If your first name happens to be George, replace it with "Abraham.")
5. Save the file but do not exit vi.
6. Without exiting vi, temporarily execute the ls command to confirm that the file was saved.
7. Enter the command that causes line numbers to appear.
8. Enter the command that causes lines 1 through 3 to be deleted.
9. Move to the line of text that contains your last name, cut it, and place the text on the clipboard.
10. Paste your last name at the beginning of the first line in the file.

LESSON B

objectives

After studying this lesson, you should be able to:

- Compare and contrast the features of Emacs and the vi editor
- Become familiar with the most important Emacs editor commands
- Create and edit simple documents using the Emacs editor

The Emacs Editor

Using the Emacs Editor

Emacs is another UNIX text editor that has gained popularity but, due to its complexity, continues to run second to the vi editor. Unlike vi, Emacs is not modal: it does not switch from command mode to insert mode. This means that you can type a command without verifying that you are in the proper mode. Although Emacs is more complex than vi, it is more consistent. For example, you can enter most commands by pressing Alt or Ctrl key combinations.

Emacs also supports a sophisticated macro language. A **macro** is a set of commands that automates a complex task. Think of a macro as a "super instruction." Emacs also has a powerful command syntax and extensions. Its packaged set of customized macros lets you read electronic mail and news, and edit the contents of directories. Emacs is reputed to have more features than any other UNIX program. You can start learning Emacs by learning its common commands. Table 3-3 lists these commands.

Alt command	Purpose	Ctrl command	Purpose
Alt+<	Move cursor to start of file	Ctrl+@	Mark the cursor location. After moving the cursor, you can move or copy text to the mark.
Alt+>	Move cursor to end of file	Ctrl+A	Move cursor to start of line
Alt+B	Move cursor back one word	Ctrl+B	Move cursor back one character
Alt+D	Delete current word	Ctrl+D	Delete the character under cursor

Table 3-3: Common Emacs commands

Alt command	Purpose	Ctrl command	Purpose
Alt+F	Move cursor forward one character	Ctrl+E	Move cursor to end of line
Alt+Q	Reformat current paragraph using word wrap so that lines are full	Ctrl+F	Move cursor forward one character
Alt+T	If the cursor is under the first character of the word, transposes word with the preceding word	Ctrl+G	Cancel the current command
Alt+U	Capitalize all letters of the current word	Ctrl+H	Use online help
Alt+W	Scroll up one screen	Ctrl+K	Delete text to the end of the line
Alt+X doctor	Enter doctormode to play a game in which Emacs responds to your statements with questions. Save yourwork first. Not all versions support this mode.	Ctrl+N	Move cursor to next line
Ctrl key combination		Ctrl+P	Move cursor to preceding line
Ctrl+H+C	Display the command that runs when you press a particular key	Ctrl+T	Transpose the character before the cursor and the character under the cursor
Ctrl+H+T	Run a tutorial about Emacs	Ctrl+V	Scroll down one screen
Ctrl+X, Ctrl+C	Exit Emacs	Ctrl+W	Delete marked text. Press Ctrl+Y to restore deleted text.
Ctrl+X, Ctrl+S	Save the file	Ctrl+Y	Insert text from the file buffer, and place it after the cursor
Ctrl+X, U	Undo the last change		
Ctrl+Del	Delete the character under the cursor		

Table 3-3: Common Emacs commands (*continued*)

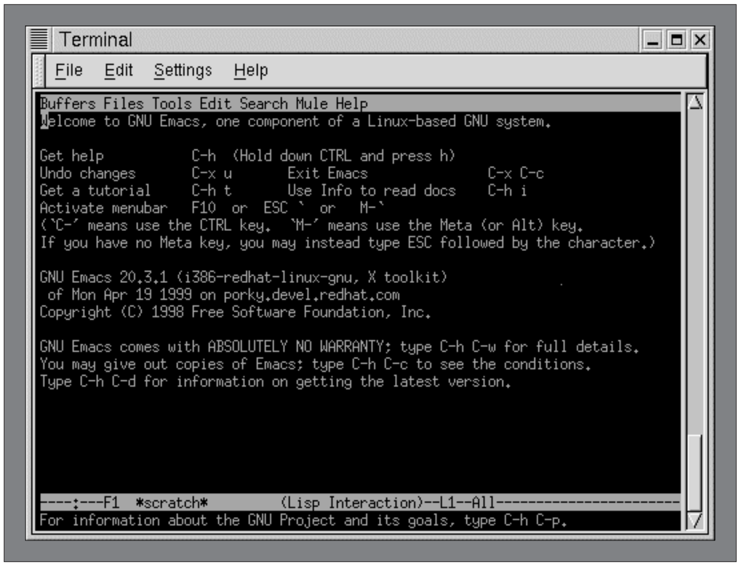
Note: In most cases, Ctrl and Alt commands in Emacs are not case-sensitive, so Alt+B and Alt+b are the same command.

You can use Emacs to duplicate your first memo for Dominion Consulting. To do so, you complete the following tasks:

- Create a new file in Emacs
- Edit and delete text
- Copy, cut, and paste text

Creating a New File in Emacs

You can start Emacs by typing the emacs command. If you type a filename after this command, Emacs creates a new, blank file with that name, or opens an existing file with that name. If you type emacs with no filename, Emacs displays the introductory list of a few important commands, shown in Figure 3-12.



```
Buffers Files Tools Edit Search Mule Help
Welcome to GNU Emacs, one component of a Linux-based GNU system.

Get help          C-h  (Hold down CTRL and press h)
Undo changes      C-x u       Exit Emacs               C-x C-c
Get a tutorial    C-h t       Use Info to read docs    C-h i
Activate menubar  F10  or  ESC `  or   M-`
(`C-' means use the CTRL key.  `M-' means use the Meta (or Alt) key.
If you have no Meta key, you may instead type ESC followed by the character.)

GNU Emacs 20.3.1 (i386-redhat-linux-gnu, X toolkit)
 of Mon Apr 19 1999 on porky.devel.redhat.com
Copyright (C) 1998 Free Software Foundation, Inc.

GNU Emacs comes with ABSOLUTELY NO WARRANTY; type C-h C-w for full details.
You may give out copies of Emacs; type C-h C-c to see the conditions.
Type C-h C-d for information on getting the latest version.



----:---F1  *scratch*        (Lisp Interaction)--L1--All----------------------
For information about the GNU Project and its goals, type C-h C-p.
```

Figure 3-12: Emacs opening screen (without a filename)

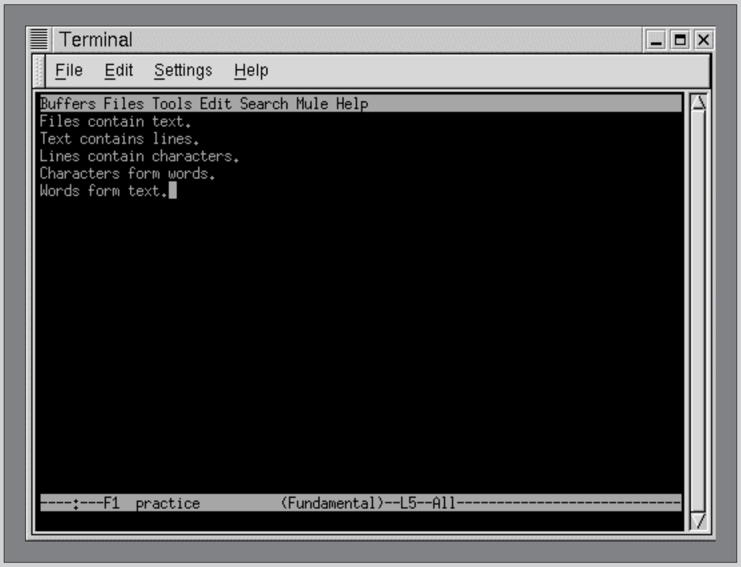
Start by creating a file called practice that will contain the same text as your original temp memo for Dominion Consulting.

To start Emacs and create a file called practice:

1 Type **emacs practice.fil** and then press **Enter**.

You see the opening screen; its status bar indicates you are creating a new file.

2 To add text to the file, type the text you see in Figure 3-13.



```
Files contain text.
Text contains lines.
Lines contain characters.
Characters form words.
Words form text.
----:---F1  practice         (Fundamental)--L5--All--------------------------
```

Figure 3-13: Creating a new file in Emacs

3 Press **Ctrl+x** and then **Ctrl+s** to save the file.

4 Press **Ctrl+x** and then **Ctrl+c** to exit the file.

Note: In Emacs, you must press the two Ctrl key combinations to save and exit a file.

Editing an Emacs File

To navigate an Emacs file, you can use either the cursor movement keys—such as the arrow keys, Pg up, Pg down, Home, and End—or Ctrl key combinations. Before editing practice, move around the file.

To navigate the practice file:

1 To return to Emacs and retrieve your file, type **Emacs practice.fil** and press **Enter**.

2 Press **Ctrl+f** to move forward one character, and then press the **right arrow** key.

3 Press **Ctrl+b** to move back one character, and then press the **left arrow** key.
4 Using the **up arrow** and **down arrow** keys, place the cursor on the line that begins "Lines contain."

To delete text and undo the deletion:

1 Press **Ctrl+k** to delete the current line.
2 Press **Ctrl+x** and then type **u** to undo the last change.

tip

You can restore the text repeatedly, even after making many changes. Press Ctrl+x, and then u. Use this command to undo your editing commands in sequence. The Emacs undo command offers an advantage over the vi undo command, which only restores your most recent change.

In Emacs, you can insert text simply by typing. You can also insert text by copying and pasting or cutting and pasting. You can copy the first two lines of the practice file and paste them at the end of the file.

To copy and paste text in Emacs:

1 Move the cursor to the beginning of the sentence "Characters form words."
2 Press **Ctrl+spacebar**. This marks the starting point for the block of text you want to copy. You see the words "Mark set" in the status bar.
3 Press the **down arrow** key twice to move the cursor to the next line.
4 Press **Alt+w**. This marks the end of the text block to copy.
  You can also hold down Esc and press w to mark the end of the block.
5 Press **Alt+>** to move to the end of the file.
6 Press **Ctrl+y** to paste the marked text from the clipboard into the buffer.

Performance tip

To cut and paste rather than copy and paste, press Ctrl+w when the cursor is at the end of the block you want cut.

Like the vi editor, Emacs lets you search for specific text. For example, suppose you want to find the text "on" in your practice document.

To search for specific words:

1. Press **Ctrl+s**. You see the "I-search" prompt in the status line. You can now type the text you are seeking.
2. Type **it**.
3. Press **Ctrl+s** to search for the next occurrence of "on." Press **Ctrl+s** again.
4. Press **Ctrl+r** to search backward for the previous occurrence of "on."

To reformat the document so lines are full with text:

1. The first and third lines each have the word "contain." On each line, move the cursor to the space after the word "contain" and press **Enter**. This breaks each of the lines into two lines.
2. Press **Alt+Q** to reformat the file so the lines are full of text.

After working with the practice document, you're ready to exit Emacs. To do so, follow the instructions in the earlier section, "Creating a New File in Emacs." (Press Ctrl+x and then press Ctrl+c to save and exit the file.)

S U M M A R Y

- The Emacs editor has gained popularity, but due to its complexity, it continues to run second to the vi editor.
- Unlike vi, Emacs is not modal. (It does not switch from command mode to insert mode.)
- Emacs has a powerful command syntax and extensions, and supports a sophisticated language of macro commands. A macro is a set of commands designed to simplify a complex task. Emacs' packaged set of customized macros lets you read electronic mail and news, and edit the contents of directories.
- You can start Emacs by typing emacs with or without a filename. If you enter this command and then type a filename, Emacs creates a new, blank file with that name, or opens an existing file with that name. If you type emacs with no filename, Emacs displays an introductory list of a few important commands.
- You can use either the cursor movement keys—such as the arrow keys, Pg up, Pg down, Home, and End—or Ctrl key combinations to navigate an Emacs file.
- You can undo your editing changes in sequence, even after you've made many changes. The Emacs undo command offers an advantage over the vi undo command, which can only restore your most recent change.

- In Emacs, you can insert text simply by typing. You can also insert text by copying and pasting or cutting and pasting. Like the vi editor, Emacs also lets you search for specific text.

COMMAND SUMMARY

Chapter 3, Lesson B commands

Command	Purpose
Emacs Commands	See Table 3-3
emacs	Start the Emacs editor
UNIX Commands	
lpr	Print a file

Note: Table 3-3 lists common Emacs commands.

REVIEW QUESTIONS

1. The Emacs editor ________________.
a. is modeless
b. is modal
c. is capable of both modal and modeless operations
d. has insert and command modes

2. To issue a command in Emacs, press ________________.
a. down the Ctrl key, and then press a letter
b. down the Alt key, and then press a letter
c. the function keys (F1–F12)
d. a and b

3. Which of these Ctrl key combinations lets you quit the Emacs editor and return to the command line?
a. Ctrl+c, Ctrl+x
b. Ctrl+x, Ctrl+c
c. Ctrl+s, Ctrl+x
d. Ctrl+x, Ctrl+e

4. Which of these Ctrl key combinations saves your file?
   a. Ctrl+c, Ctrl+x
   b. Ctrl+x, Ctrl+y
   c. Ctrl+x, Ctrl+s
   d. Ctrl+z, Ctrl+s

5. Which of these best compares Emacs to the vi editor?
   a. Emacs is popular, but due to its complexity, continues to run second to the vi editor.
   b. The vi editor has a sophisticated macro language, but Emacs does not.
   c. Both have an insert mode and a command mode.
   d. All of the above are correct.

6. What do you see when you use the Emacs command with no filename to start editing an Emacs file?
   a. A blank screen for an unnamed file
   b. A list of common Emacs commands
   c. The text of the last file you edited
   d. A summary of differences between Emacs and the vi editors

7. Which of these is false?
   a. In Emacs, you can press Del to delete characters.
   b. In Emacs, you can use all the arrow keys to move up, down, left, or right.
   c. In Emacs, you can use Ctrl+h to move the cursor back one space.
   d. In Emacs, you can use Ctrl+a to move text to the beginning of the line.

8. True or false: In most cases, Ctrl and Alt commands in Emacs are not case-sensitive.

9. The ______________ command marks the start of the block of text you want to copy.
   a. Ctrl+M
   b. Ctrl+S
   c. Ctrl+spacebar
   d. none of the above

10. The ______________ command marks the end of the block of text you want to copy.
   a. Alt+w
   b. Ctrl+w
   c. Alt+spacebar
   d. none of the above

11. The ______________ command moves the cursor to the end of the file.
   a. Ctrl+<
   b. Ctrl+>
   c. Alt+<
   d. Alt+>

12. The ______________ command inserts text from the clipboard and places it after the cursor.
   a. Ctrl+p
   b. Ctrl+y
   c. Alt+p
   d. Alt+y

13. The ______________ command is used to undo the last command.
a. Ctrl+u
b. Ctrl+x
c. Ctrl+u,x
d. Ctrl+x,u

14. The ______________ command causes the current line to be deleted.
a. Ctrl+k
b. Ctrl+l
c. Ctrl+x,k
d. Alt+k

15. The ______________ command invokes the I-search prompt.
a. Ctrl+s
b. Alt+s
c. Ctrl+spacebar
d. none of the above

16. The ______________ command searches backward.
a. Ctrl+x,b
b. Ctrl+b
c. Ctrl+r
d. none of the above

17. True or false: The Emacs undo command is like the vi undo command. Both can only undo the last command.

EXERCISES

Use Table 3-3 to find the correct commands for performing the following steps.

1. Using the Emacs editor, create a new file that contains the first 4 lines of Shakespeare's 80th Sonnet:
O, how I faint when I of you do write,
Knowing a better spirit doth use your name,
And in the praise thereof spends all his might,
To make me tongue-tied, speaking of your fame!

2. Move the cursor to any letter except "b" in the word "better" on the second line.

3. Use the command that causes the current word to be transposed with the one that follows it. After executing the command, the line should read:
Knowing a spirit better doth use your name,

4. Move the cursor to the word "doth" on the same line.

5. Use the command to delete the current word.

6. Move the cursor to the first character of the word "spirit" on the same line.

7. Use the command that capitalizes the letters of the word.

8. Move the cursor under the letter "y" in the word "your" on the same line.
9. Use the command that deletes the character under the cursor. The line should now read:
   Knowing a SPIRIT better use our name,
10. Move the cursor to the word "spends" on the third line.
11. Use the command that deletes text to the end of the line. The line should now read:
    And in the praise thereof
12. Move the cursor to the end of the first line.
13. Use the command that puts a mark at the cursor location.
14. Move the cursor to the first character of the first line.
15. Use the command that deletes marked text, that is, deletes the first line.
16. Move the cursor to the end of the file. Use the command that restores deleted text. The text that was the first line of the file is now at the end of the file.

DISCOVERY EXERCISES

1. Using the practice file, practice copying and pasting the text to rearrange the order of the lines.
2. Add text to the file, and practice using the cursor movement commands.
3. Replace all occurrences of the word "the" with "a."
4. Select five words and convert them to all uppercase.
5. Delete a line and then undo the deletion.
6. Transpose the first two words in each line.
7. Save the file and exit Emacs.